AFRL-AFOSR-VA-TR-2016-0181

Simulation-based Model Checking for Nondeterministic Systems and Rare Events

Edmund Clarke
CARNEGIE MELLON UNIVERSITY

03/24/2016
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTA2
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
14-03-2016	FINAL	01-02-2012 - 30-09-2015

4. TITLE AND SUBTITLE
Simulation-Based Model Checking for Nondeterministic Systems and Rare Events

5a. CONTRACT NUMBER
PO 284227

5b. GRANT NUMBER
FA9550-12-1-0146

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Prof. Edmund Clarke (CMU)
Dr. David Musliner (SIFT)
Dr. Robert Goldman (SIFT)
Soonho Kong (CMU)
Qinsi Wang (CMU)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Carnegie Mellon University 5000 Forbes Ave. Pittsburgh, PA 15213

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AF Office of Scientific Research 875 North Randolph Street Arlington VA 22203–1995

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION A

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Objective: This project extends statistical model checking methods to enable reasoning about nondeterministic systems and extremely rare events.
Approach: To address nondeterministic systems, we investigated a theoretical framework integrating semi-exhaustive simulation with hierarchical abstraction of models. For rare events, we investigated importance sampling methods that rely on variance minimization and cross- entropy methods to optimize biasing distributions, allowing statistical methods to reason accurately about low-probability events.
Outcome/Impact: Statistical model checking methods scale better than traditional analytic methods for very large systems; this research is critical to allow statistical methods to reason about realistic systems involving nondeterminism and low-probability events. Our new methods will be implemented in the PRISMATIC tool, supporting testing, evaluation, and eventually application to verification of real-world systems.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE		11	Edmund Clarke
					19b. TELEPHONE NUMBER *(Include area code)* 412-268-2628

Reset

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE. Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE. State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATES COVERED. Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. TITLE. Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER. Enter all contract numbers as they appear in the report, e.g. F33615-86-C-5169.

5b. GRANT NUMBER. Enter all grant numbers as they appear in the report, e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER. Enter all program element numbers as they appear in the report, e.g. 61101A.

5d. PROJECT NUMBER. Enter all project numbers as they appear in the report, e.g. 1F665702D1257; ILIR.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER. Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER. Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

Simulation-based Model Checking for Nondeterministic Systems and Rare Events

Final Report

March 14, 2016

Project Monitor: Prof. Ed Clarke (CMU); Dr. Robert Bonneau (AFOSR)

Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
and
AF Office of Scientific Research
875 North Randolph Street
Arlington VA 22203–1995

Contract Number PO 284227 under CMU prime contract FA9550-12-1-0146

1 Project Summary

Objective: This project extends statistical model checking methods to enable reasoning about nondeterministic systems and extremely rare events.

Approach: To address nondeterministic systems, we investigated a theoretical framework integrating semi-exhaustive simulation with hierarchical abstraction of models. For rare events, we investigated importance sampling methods that rely on variance minimization and cross-entropy methods to optimize biasing distributions, allowing statistical methods to reason accurately about low-probability events.

Outcome/Impact: Statistical model checking methods scale better than traditional analytic methods for very large systems; this research is critical to allow statistical methods to reason about realistic systems involving nondeterminism and low-probability events. Our new methods will be implemented in the PRISMATIC tool, supporting testing, evaluation, and eventually application to verification of real-world systems.

2 Statistical Model Checking for Markov Decision Processes

We have been investigating the use of techniques for model-checking systems described as probabilistic automata that have both statistical elements and pure (unquantified) nondeterminism.

Typically, we model-check safety assertions in bounded-time LTL (BLTL), such as $P_{t\leq 10}(\phi) < P$ – the probability that ϕ will occur in 10 time steps or less is less than P, where ϕ would be some undesirable property such as system failure, deadlock, etc. The problem of model-checking such systems is equivalent to solving a Markov Decision Problem (MDP), where we must show that no matter how an adversary resolves the non-deterministic choices, the probability of ϕ must be less than P; *i.e.*, that the system description forces a win against a (nondeterministic) adversary who is trying to make ϕ occur with a probability of at least P.

These model-checking problems are *extremely* computationally demanding, both in terms of time and space. CMU researchers have addressed this problem with their development of Statistical Model Checking for Markov Decision Processes (SMCMDP) [3]. The SMCMDP algorithm operates in two phases: the first phase uses learning to find the worst-case policy for the nondetermistic adversary, and the second phase uses that learned policy to simplify the problem and solve for a model-checking conclusion. The first phase resolves the nondeterminism with an initial probability distribution, and then uses multiple rounds of Monte Carlo sampling and Reinforcement Learning to improve the policy for nondeterminic choices with respect to satisfying a Bounded Linear Temporal Logic (BLTL) property. The second phase uses the best learned policy to reduce an MDP to a fully probabilistic Markov chain, on which known statistical model checking methods may be applied to give an approximate solution to the problem of checking the probabilistic BLTL property.

In the last year, we have investigated AO* search and Monte Carlo Tree Search algorithms to complement and enhance CMU's SMCMDP.

2.1 Challenges in Statistical Model Checking for Markov Decision Processes

CMU's SMCMDP implementation can substantially ease the runtime requirements of our model-checking problems. It easily adapts to parallel and multi-core systems, providing further speedups. Nevertheless, challenges remain, in particular:

- While the SMCMDP technique soundly demonstrates property violations (where the probability of ϕ exceeds the desired value), it cannot accurately identify cases where the property is necessarily *satisfied*.
- In order to use efficient MDP-solving techniques, SMCMDP can derive only *memoryless, stationary* policies for the adversary. This can compromise identification of violations, in cases where time considerations are necessary to force the system to violate the property with the required probability. In other words, SMCMDP only reasons about a limited memoryless form of adversary, but there can be situations where stateful adversaries are more hazardous, so the SMCMDP assurances are not correct, in general.
- When there are extreme probabilities in the models, sampling in SMCMDP converges slowly.

2.2 AO* Search

We have been developing methods to check bounded-time properties of probabilistic automata using heuristic search. The strengths (and weaknesses) of heuristic search nicely complement those of sampling methods and dynamic programming (as in PRISM). In particular, when the heuristic performs well, we can avoid enumerating the full state space. Like dynamic programming, but unlike statistical methods, AO* search guarantees the correctness of the probability bounds it computes. This is particularly important when we *fail* to find a counterexample for a claim: if we report that a system is safe because it *cannot* reach a safety-violating state, s, with greater than a probability P, we can make this claim with confidence. Since the sampling methods do not currently provide bounds on the quality of the policy (counterexample) they compute, we cannot currently make such safety claims with confidence.[1]

Search algorithm: We have implemented a version of AO* search as an extension to the PRISM probabilistic model checker. AO* search explores an AND/OR tree, so we can use it to find the probability of reachability for a property in PRISM's Probabilistic LTL. By finding the maximum probability of reachability, we can check properties of the form "what is the maximum probability of reaching a state that satisfies ϕ in less than k steps?" The problem is an AND/OR search, rather

[1] We can be probabilistically certain that we cannot reach s with a probability greater than P, based on the adversary policy chosen by sampling, but we cannot currently provide informative guarantees that the adversary policy is optimal, or close enough to optimal to justify the safety claim.

than a simple graph reachability, because we must compute the best choice for reachability (OR) for all of the possible outcomes of the probabilistic branches (AND).

Our initial implementation was based on the text by Edelkamp and Schrödl [2]. We were hampered by a substantial error in the book's presentation of the algorithm. We have reported that error to the authors; an erratum will be prepared. In addition, we made substantial modifications to the algorithm in order to make it (1) interface with the methods and data structures of the PRISM model checker, (2) incorporate an informed heuristic (see below), and (3) exploit special features of the MDP search problems.

Heuristic: In order to get acceptable performance from AO*, we must have an *informed* and *admissible* heuristic. The heuristic must be admissible, or we cannot ensure that we will find the optimal probability of reachability. We have used heuristics inspired by methods that have been found effective in AI planning. We initially experimented with simple reachability, simply distinguishing between states from which the goal is and is not reachable. This could be efficiently computed, using BDDs, but was not sufficiently informative. We replaced this "disjunctive" heuristic with a "metric" heuristic that computes an (over)estimate of the probability of reaching the goal from each state. We can compute this efficiently by doing a backwards reachability computation from the goal state, implemented using ADDs. An exact backwards reachability computation is the dynamic programming method used by PRISM, so for efficient computation, we must relax this computation to an over-estimate (over- for admissibility). We do this by quantifying away the action decisions, effectively acting as if we could take *all* the decision from an OR node. In our experiments so far, this heuristic estimate provides a good compromise between information content and efficient computability (see preliminary example below). Further abstraction in the heuristic may prove necessary as we experiment with more models.

Performance of Search Algorithm: As an example, see Figure 1, which shows a preliminary comparison between our AO* search algorithm and PRISM on a scaled set of WLAN examples. The WLAN examples were taken from the PRISM web page (`http://www.prismmodelchecker.org/casestudies/wlan.php`). This model describes the handshake and randomized exponential backoff rules used for collision avoidance in the IEEE 802.11 standard [5]. For more details on the PRISM modeling and verification, see [5]. We scale the problems by extending the length of the permissible back-off in the face of collisions. As will be seen in the PRISM results, this causes the state space, and hence run time, to grow. On the other hand, the AO* search, with its heuristic guidance, is not sensitive to the growth in the state space. We will examine different models to identify which are most suitable for which solution methods (search, sampling, dynamic programming).

As we perform more comparisons, we expect to find weaknesses in the AO* implementation and make improvements. For example, recent tests of the performance and correctness of the algorithm

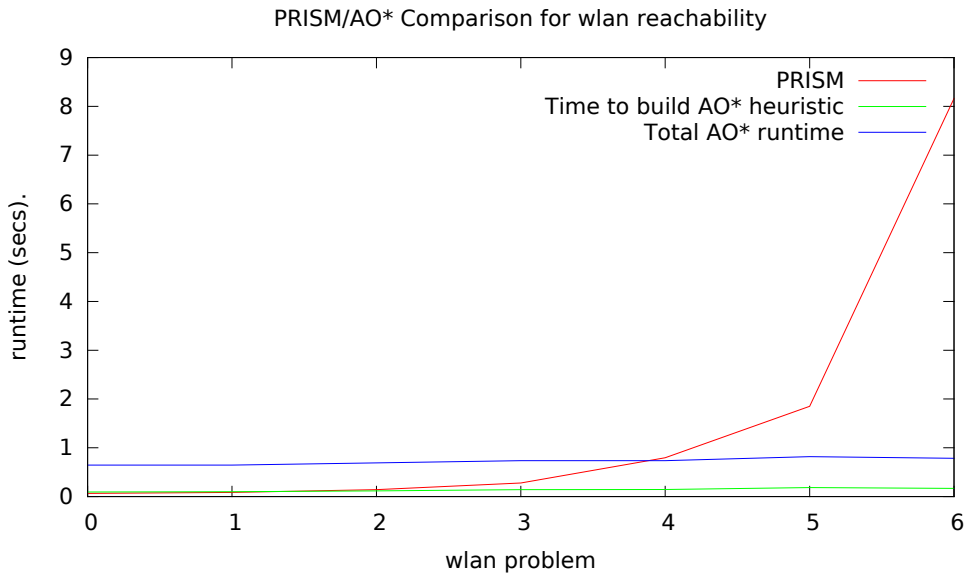

Figure 1: Comparison between AO* algorithm and PRISM's dynamic programming on scaled WLAN problems.

revealed inefficient data structures for representing the best partial solution in the search. Improving these data structures provided a substantial speedup.

We are also investigating whether to extend our AO* search to AO* *branch-and-bound* search [6, 7]. Branch-and-bound might allow us to prune substantial parts of the search space, providing memory savings, particularly when handling very large models.

2.3 Monte Carlo Tree Search

The Monte Carlo sampling process in SMCMDP can take a long time to converge. This problem can manifest itself either in the first phase where reinforcement learning is used to find an adversary policy (resolving non-determinism in the model), or in the second phase when, after the non-determinism has been resolved, we sample from the resulting Markov Chain to evaluate the BLTL property's worst case probability.

To improve performance in the first phase of SMCMDP, SIFT has been experimenting with Monte Carlo Tree Search (MCTS) methods [1]. These methods have been very successful in difficult search applications, including Computer Go, and planning under uncertainty. SIFT has developed two sampling methods using the Upper Confidence Bounds Applied to Trees (UCT) Monte Carlo Tree Search algorithm [4]: *offline UCT* which computes a policy over the full state space, and *online UCT* which estimates the probability of the property against the optimal adversary.

Model	States	True Probability	Threshold Probability (P)	Learning Samples	Correct w/o UCT	Correct w/ UCT
CSMA 2 2	1038	0.861	0.85	2000	12%	**77%**
CSMA 2 4	7958	0.768	0.76	2000	18%	**70%**
CSMA 2 6	66718	0.616	0.61	2000	25%	**54%**
CSMA 2 6	66718	0.616	0.61	4000	22%	**74%**

Figure 2: Initial results of UCT-guided SMCMDP on different-sized models of the probabilistic CSMA protocol. For each model, we asked SMCMDP and SMCMDP with UCT-guided sampling: using the given number of samples, is the threshold probability less than the true probability. The percentage correct is out of 100 runs.

Offline UCT: Our offline UCT method simply replaces the sampling method in SMCMDP with UCT. We expected that UCT's nice property of balancing exploitation of known good actions with exploration of seldom explored actions would lead to finding an optimal (or near-optimal) adversary policy with fewer samples. We have implemented offline UCT as an extension to PRISMATIC.

Our initial experiments with offline UCT looked promising. As seen in Figure 2, offline UCT yielded the correct answer much more often than "vanilla" SMCMDP, learning from the same number of samples, for difficult problems where the threshold probability is very close to the true probability. This indicated that UCT helped SMCMDP learn a *better policy* with the *same number of samples*.

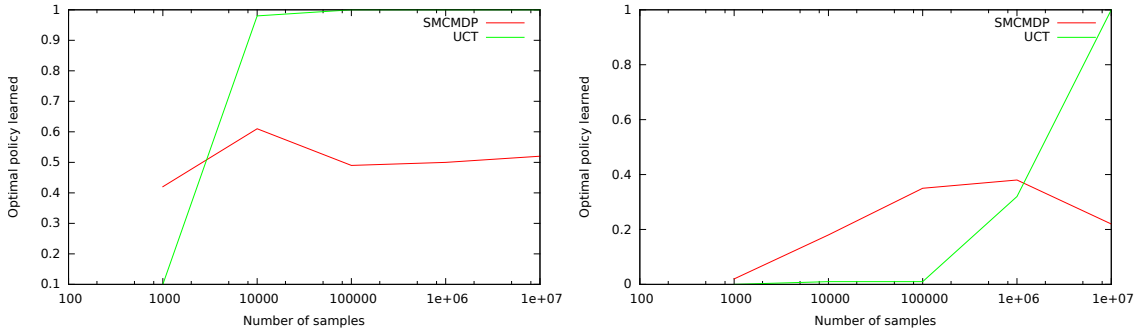

Figure 3: How often offline UCT and SMCMDP learn the correct policy for an easier (left) and more difficult (right) version of our satellite model.

Next, we compared how many samples it took SMCMDP and offline UCT to learn the optimal adversarial policy for a scalable satellite control model we created. As seen in Figure 3, offline UCT converges to the optimal policy, but with the default parameters SMCMDP did not converge to the optimal policy. This model is very small (15 states, 52 transitions, 30 actions), so it was

discouraging to see how many traces it took to find the optimal policy.

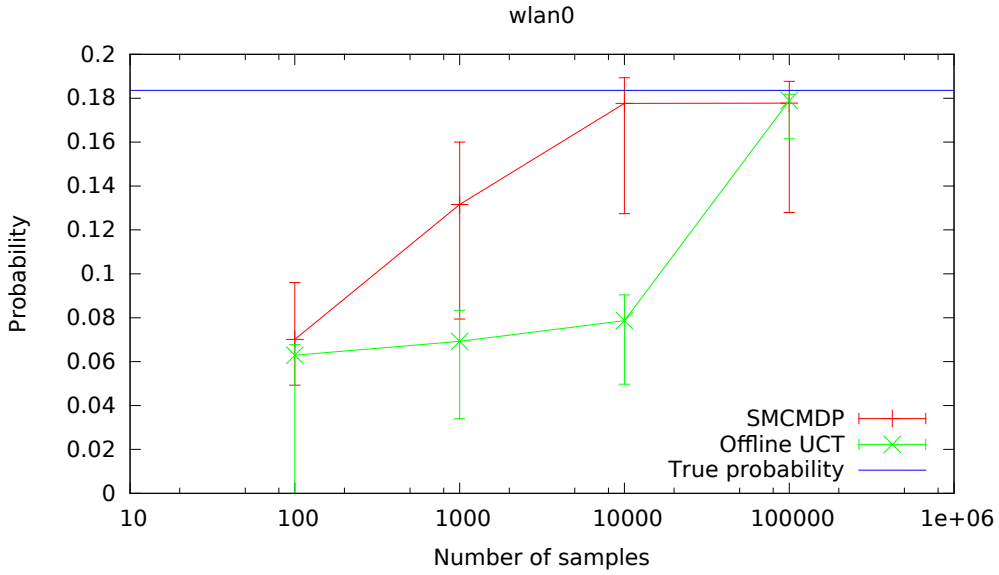

Figure 4: Probability estimates from SMCMDP on a WLAN model, the averaged over 100 runs.

We also ran experiments on some larger models, where we do not have the (very large) optimal policy readily available, but we do have the true underlying probability of the property. To evaluate offline UCT on these models, we had it learn the policy, then estimate the probability of the property using that policy. These experiments show that the quality of offline UCT's policy lags behind SMCMDP with a small number of traces, but does catch up with more traces (see Figure 4).

We believe offline UCT's need for more samples than SMCMDP results from a difference in bookkeeping between the two algorithms. When SMCMDP takes a trace, it remembers the reward in every state along that trace. When UCT takes a trace, it only remembers the rewards for states that have been expanded in its search tree. We will see if modifying offline UCT to remember rewards along all states in every trace will reduce the number of traces required to learn a good policy, at the cost of using more memory.

Online UCT: The typical application of UCT to playing a game (like Go) does not involve computing a policy for the entire state space, as our offline UCT algorithm attempts. Rather, it runs online—that is, it only takes one action at a time (a move), then senses the opponent's move, and repeats until the game finishes. Our online UCT algorithm follows this game-playing analogy, with our moves being the resolution of non-determinism, the opponent's moves being the resulting probabilistic transitions, and a game is won if we satisfy the property and lost if we do not.

In this framework, each "game" results in one trace through the system, using the best-looking action from each state as determined by UCT sampling. After playing many games, we can look

Problem size	0	1	2	3	4	5	6
Probability est.	0.187	0.166	0.166	0.172	0.187	0.170	0.189
Relative Error	1.9%	9.4%	9.7%	6.6%	2.1%	7.2%	2.9%

Figure 5: Online UCT probability estimates after 800 games, with 1000 samples per move on WLAN models. The models range from 6,063 to 5,007,666 states. The true underlying probability is about 0.184.

at how often the property is satisfied to get an estimate of the probability of the property.

We have implemented online UCT as an extension to PRISMATIC, and started running experiments. Figure 5 shows online UCT's probability estimates for the set of WLAN models from PRISM's case studies. All estimates are under 10% error, with a small number of games relative to the size of the state space. Also note the quality of the probability estimates is not sensitive to the size of the model's state space, indicating that online UCT focuses on the interesting parts of the state space.

Currently the number of games to play is given as input. We plan to use Wald's Sequential Probability Ratio Test (SPRT) as a termination criteria, which will minimize the number of games played to achieve a given level of confidence in our answer [8]. Likewise, the number of samples taken prior to making each move is given as input; we will investigate optimizing this as well.

Another opportunity for improvement is to share information between games. The games online UCT plays are all currently independent; that is, none of the sampling results from one game are carried to future games. Carrying forward some data between games could improve the results. We could, for example, keep a cache of frequently-encountered states, or learn an "opening book" of good moves near the initial state.

2.4 Time-Dependent Policies

One limitation of CMU's SMCMDP is that it produces only stationary, memoryless policies (adversaries). But when performing *bounded time* model-checking, in general the optimal adversary policy is time-dependent. That means that a model-checker using a stationary policy may incorrectly label some systems as safe. The challenge of finding time-dependent policies is twofold: (1) keeping track of time in the model causes state space explosion, and (2) in general, a separate policy is required for every time unit, making the policy very large.

We have developed several test problems for experimenting with time-dependent vs. stationary adversaries. For the simple example in Figure 6, each transition takes one time tick, and the adversary wins by driving the model into `fail`. He starts at `s0` and must choose action `L` or `S`; after that the transitions are determined by the indicated probabilities. With 4 or more time ticks left, `L` has the highest probability of failure. However, with 3 time ticks left, `fail` is unreachable

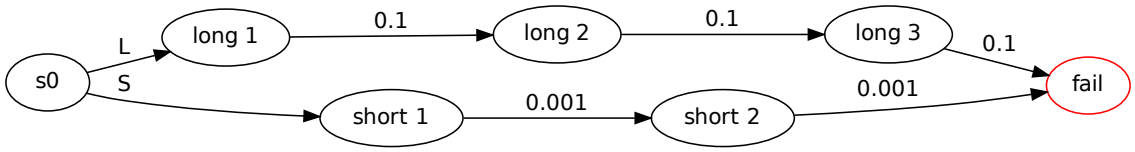

Figure 6: A simple example where the optimal adversary requires a time-dependent policy.

through `L`, so the adversary should choose action `S`.

We have begun exploring methods for efficiently developing time-dependent adversaries. One possibility, suggested by our adoption of UCT, is to compute the adversary's "moves" on-line, avoiding the need to store full policies. Other techniques we are considering include exploiting structured state models, and compressing large policies.

3 Delta-complete Analysis

We have developed the framework of delta-complete analysis [9] for bounded reachability problems of general hybrid systems. We perform bounded reachability checking through solving delta-decision problems over the reals. The techniques take into account of robustness properties of the systems under numerical perturbations. We prove that the verification problems become much more mathematically tractable in this new framework. Our implementation of the techniques, an open-source tool dReach, scales well on several highly nonlinear hybrid system models that arise in biomedical and robotics applications. We developed a framework to give upper bounds on the computational complexity of stability problems for a wide range of nonlinear continuous and hybrid systems. To do so, we describe stability properties of dynamical systems using first-order formulas over the real numbers, and reduce stability problems to the delta-decision problems of these formulas. The framework allows us to obtain a precise characterization of the complexity of different notions of stability for nonlinear continuous and hybrid systems. We proved that bounded versions of the stability problems are generally decidable, and give upper bounds on their complexity. The unbounded versions are generally undecidable, for which we give upper bounds on their degrees of unsolvability.

We developed a novel approach for solving the probabilistic bounded reachability problem of hybrid systems with parameter uncertainty [10]. Standard approaches to this problem require numerical solutions for large optimization problems, and become unfeasible for systems involving nonlinear dynamics over the reals. Our approach combines randomized sampling of probabilistic system parameters, SMT-based bounded reachability analysis, and statistical tests. We utilize delta-complete decision procedures to solve reachability analysis in a sound way, i.e., we always decide correctly if, for a given combination of parameters, the system actually reaches the unsafe

region. Compared to standard simulation-based analysis methods, our approach supports non-deterministic branching, increases the coverage of simulation, and avoids the zero-crossing problem. We demonstrate that our method is feasible for general hybrid systems with parametric uncertainty by applying the implemented tool - SReach - to various nonlinear hybrid systems with parametric uncertainty.

We found serious bugs in floating-point computations for evaluating elementary functions in the Embedded GNU C Library [11]. For instance, the sine function can return values larger than 1053 in certain rounding modes. Further investigation also exposed faulty implementations in the most recent version of the library, which seemingly fixed some bugs, but only by discarding user-specified rounding-mode requirements. We discuss our experience in how these bugs were spotted and how they affected the implementation process of our SMT solver dReal.

4 Acknowledgments

This material is based upon work supported by the AFRL under Contract No. PO 284227 under CMU prime contract FA9550-12-1-0146. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the AFRL.

References

[1] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *Computational Intelligence and AI in Games, IEEE Transactions on*, vol. 4, no. 1, pp. 1–43, 2012.

[2] S. Edelkamp and S. Schrödl, *Heuristic Search - Theory and Applications*, Academic Press, 2012. Available online as `http://www.elsevierdirect.com/product.jsp?isbn=9780123725127.\newblock`.

[3] D. Henriques, J. Martins, P. Zuliani, A. Platzer, and E. Clarke, "Statistical Model Checking for Markov Decision Processes," *9th International Conference on Quantitative Evaluation of SysTems (QEST)*, 2012.

[4] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *Machine Learning: ECML 2006*, pp. 282–293, Springer, 2006.

[5] M. Kwiatkowska, G. Norman, and J. Sproston, "Probabilistic Model Checking of the IEEE 802.11 Wireless Local Area Network Protocol," in *Proc. 2nd Joint International Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification*

(PAPM/PROBMIV'02), H. Hermanns and R. Segala, editors, volume 2399 of *LNCS*, pp. 169–187. Springer, 2002.

[6] R. Marinescu and R. Dechter, "AND/OR Branch-and-Bound Search for Combinatorial Optimization in Graphical Models," *Artificial Intelligence*, vol. 173, no. 16–17, pp. 1457–1491, 2009.

[7] L. Otten, *Extending the Reach of AND/OR Search for Optimization in Graphical Models.*, PhD thesis, UCLA, 2013.

[8] A. Wald, "Sequential Tests of Statistical Hypotheses," *The Annals of Mathematical Statistics*, vol. 16, no. 2, pp. pp. 117–186, 1945. Available online as `http://www.jstor.org/stable/2235829`.

[9] S. Kong, S. Gao, W. Chen, and E. Clarke, "dReach: Delta-Reachability Analysis for Hybrid Systems," *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2015.

[10] F. Shmarov, and P. Zuliani, "ProbReach: verified probabilistic delta-reachability for stochastic hybrid systems" *International Conference on Hybrid Systems: Computation and Control*, 2015.

[11] S. Kong, S. Gao, and E. Clarke, "Floating-Point Bugs in the Embedded GNU C Library", CMU SCS Technical Report CMU-CS-13-130, 2013.

F. Shmarov, and P. Zuliani, "ProbReach: verified probabilistic delta-reachability for stochastic hybrid systems" *International Conference on Hybrid Systems: Computation and Control*, 2015.

1.

1. Report Type

Final Report

Primary Contact E-mail

Contact email if there is a problem with the report.

emc@cs.cmu.edu

Primary Contact Phone Number

Contact phone number if there is a problem with the report

412-268-2628

Organization / Institution name

Carnegie Mellon University

Grant/Contract Title

The full title of the funded effort.

Simulation-based Model Checking for Nondetermistic Systems and Rare Events

Grant/Contract Number

AFOSR assigned control number. It must begin with "FA9550" or "F49620" or "FA2386".

FA9550-12-1-0146

Principal Investigator Name

The full name of the principal investigator on the grant or contract.

Edmund Clarke

Program Manager

The AFOSR Program Manager currently assigned to the award

Robert Bonneau

Reporting Period Start Date

02/01/2012

Reporting Period End Date

09/30/2015

Abstract

Objective: This project extends statistical model checking methods to enable reasoning about nondeterministic systems and extremely rare events.
Approach: To address nondeterministic systems, we investigated a theoretical framework integrating semi-exhaustive simulation with hierarchical abstraction of models. For rare events, we investigated importance sampling methods that rely on variance minimization and cross- entropy methods to optimize biasing distributions, allowing statistical methods to reason accurately about low-probability events.
Outcome/Impact: Statistical model checking methods scale better than traditional analytic methods for very large systems; this research is critical to allow statistical methods to reason about realistic systems involving nondeterminism and low-probability events. Our new methods will be implemented in the PRISMATIC tool, supporting testing, evaluation, and eventually application to verification of real-world systems.

Distribution Statement

This is block 12 on the SF298 form.

Distribution A - Approved for Public Release

Explanation for Distribution Statement

If this is not approved for public release, please provide a short explanation. E.g., contains proprietary information.

SF298 Form

Please attach your SF298 form. A blank SF298 can be found here. Please do not password protect or secure the PDF The maximum file size for an SF298 is 50MB.

SF298.pdf

Upload the Report Document. File must be a PDF. Please do not password protect or secure the PDF . The maximum file size for the Report Document is 50MB.

main.pdf

Upload a Report Document, if any. The maximum file size for the Report Document is 50MB.

Archival Publications (published) during reporting period:

Robert P. Goldman, David J. Musliner, Michael W. Boldt, "Heuristic Search for Bounded Model Checking of Probabilistic Automata" in Proc. Int'l Workshop on Design and Implementation of Formal Tools and Systems (DIFTS), September 2015.

Michael W. Boldt, Robert P. Goldman, David J. Musliner, "Offline Monte Carlo Tree Search for Statistical Model Checking of Markov Decision Processes" in Proc. Int'l Workshop on Design and Implementation of Formal Tools and Systems (DIFTS), September 2015.

Anvesh Komuravelli, "Compositional Verification with Abstraction, Learning, and SAT Solving", Ph.D. Thesis, Carnegie Mellon University, 2015.

Qinsi Wang, Paolo Zuliani, Soonho Kong, Sicun Gao and Edmund Clarke, "SReach: A Probabilistic Bounded delta-Reachability Analyzer for Stochastic Hybrid Systems", CMSB 2015: Computational Methods in Systems Biology, Nantes, France, Sep 16 - 18, 2015.

Robert P. Goldman, Michael W. Boldt, David J. Musliner, "Employing AI Techniques in Probabilistic Model Checking" in Workshop on Model Checking and AI Planning (MOCHAP), 2014.

D. Henriques, J. Martins, P. Zuliani, A. Platzer, E. M. Clarke. Statistical model checking for Markov decision processes. In QEST 2012: 9th International Conference on Quantitative Evaluation of SysTems. IEEE, pp. 84-93.

Changes in research objectives (if any):

Change in AFOSR Program Manager, if any:

Extensions granted or milestones slipped, if any:

AFOSR LRIR Number

LRIR Title

Reporting Period

Laboratory Task Manager

Program Officer

Research Objectives

Technical Summary

Funding Summary by Cost Category (by FY, $K)

	Starting FY	FY+1	FY+2
Salary			
Equipment/Facilities			
Supplies			
Total			

Report Document

Report Document - Text Analysis

Report Document - Text Analysis

Appendix Documents

2. Thank You

E-mail user

Mar 14, 2016 04:19:53 Success: Email Sent to: emc@cs.cmu.edu